### **Pemantauan Suhu dan Kelembaban Berbasis ESP32 dan Node-RED dengan Simulasi MQTT**

### **Mohammad Haryo Hammam Q**

233140700111108

## **I. RINGKASAN EKSEKUTIF**

Proyek ini merupakan pengembangan sistem pemantauan suhu dan kelembaban berbasis Internet of Things (IoT) menggunakan mikrokontroler ESP32 dan sensor DHT22. Sistem disimulasikan tanpa perangkat fisik melalui platform Wokwi dan Visual Studio Code. Data dari sensor dikirim melalui protokol MQTT menuju broker broker.emqx.io, kemudian diproses, disimpan di InfluxDB, dan divisualisasikan dalam bentuk grafis melalui dashboard Node-RED. Sistem juga mendukung komunikasi dua arah, seperti pengendalian LED dari jarak jauh. Simulasi membuktikan bahwa sistem ini responsif dan mampu menangani alur data secara real-time.

## **II. PENDAHULUAN**

### **2.1 Latar Belakang**

IoT kini menjadi teknologi kunci dalam pengembangan sistem otomatisasi, salah satunya untuk memantau parameter lingkungan seperti suhu dan kelembaban. Dengan pendekatan MQTT (Message Queuing Telemetry Transport), data dapat dikirimkan secara ringan dan efisien ke server atau broker. Visualisasi data kemudian dapat disajikan secara interaktif melalui Node-RED dan disimpan di InfluxDB sebagai basis data time-series.

Melalui simulasi ini, kita dapat memahami alur kerja sistem IoT secara menyeluruh: dari akuisisi data oleh sensor, pengiriman via jaringan, hingga penyajian visual. Hal ini memberikan pondasi penting sebelum sistem diimplementasikan secara fisik.

## **III. TUJUAN PRAKTIKUM**

* Membuat sistem monitoring suhu dan kelembaban menggunakan ESP32 dan DHT22 secara virtual.
* Mengirimkan data melalui protokol MQTT ke broker publik.
* Menyimpan dan memvisualisasikan data dengan InfluxDB dan Node-RED.
* Menguji komunikasi dua arah antara broker dan ESP32 (publish-subscribe).

## **IV. PERANGKAT DAN PERANGKAT LUNAK**

### **Perangkat Keras (Disimulasikan)**

* ESP32 Devkit V4
* Sensor DHT22
* LED indikator

### **Perangkat Lunak**

* Wokwi (IoT Simulator)
* Visual Studio Code + PlatformIO
* MQTT Broker: broker.emqx.io
* Node.js dan Node-RED
* InfluxDB (Cloud)

## **V. ARSITEKTUR SISTEM**

Diagram JSON Wokwi:

{

"version": 1,

"author": "awikwok",

"editor": "wokwi",

"parts": [

{ "type": "board-esp32-devkit-c-v4", "id": "esp", "top": 19.2, "left": -4.76, "attrs": {} },

{ "type": "wokwi-led", "id": "led1", "top": -10.8, "left": 285, "rotate": 90, "attrs": { "color": "red" } },

{ "type": "wokwi-led", "id": "led2", "top": 94.8, "left": 265.8, "rotate": 90, "attrs": { "color": "yellow" } },

{ "type": "wokwi-led", "id": "led3", "top": 46.8, "left": 275.4, "rotate": 90, "attrs": { "color": "limegreen" } },

{ "type": "wokwi-resistor", "id": "r1", "top": 32.75, "left": 163.2, "attrs": { "value": "1000" } },

{ "type": "wokwi-resistor", "id": "r2", "top": 61.55, "left": 163.2, "attrs": { "value": "1000" } },

{ "type": "wokwi-resistor", "id": "r3", "top": 90.35, "left": 163.2, "attrs": { "value": "1000" } }

],

"connections": [

[ "esp:TX", "$serialMonitor:RX", "", [] ],

[ "esp:RX", "$serialMonitor:TX", "", [] ],

[ "esp:23", "r1:1", "green", [ "h0" ] ],

[ "esp:22", "r2:1", "green", [ "h0" ] ],

[ "esp:21", "r3:1", "green", [ "h0" ] ],

[ "r1:2", "led1:A", "green", [ "v0" ] ],

[ "r2:2", "led3:A", "green", [ "v0" ] ],

[ "r3:2", "led2:A", "green", [ "v0" ] ],

[ "led1:C", "led3:C", "green", [ "h0" ] ],

[ "led3:C", "led2:C", "green", [ "h0" ] ],

[ "led2:C", "esp:GND.3", "green", [ "h0" ] ]

],

"dependencies": {}

}

VI. KODE PROGRAM ESP32

#include <WiFi.h>

#include <PubSubClient.h>

#include <DHTesp.h>

const int DHT\_PIN = 15;

const int LED\_RED = 2;

DHTesp dht;

const char\* ssid = "Wokwi-GUEST";

const char\* password = "";

const char\* mqtt\_server = "broker.emqx.io";

WiFiClient espClient;

PubSubClient client(espClient);

unsigned long lastMsg = 0;

void setup\_wifi() {

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

}

}

void callback(char\* topic, byte\* payload, unsigned int length) {

if ((char)payload[0] == '1') digitalWrite(LED\_RED, HIGH);

else digitalWrite(LED\_RED, LOW);

}

void reconnect() {

while (!client.connected()) {

String clientId = "ESP32Client-" + String(random(0xffff), HEX);

if (client.connect(clientId.c\_str())) {

client.subscribe("IOT/Test1/mqtt");

} else {

delay(5000);

}

}

}

void setup() {

Serial.begin(115200);

pinMode(LED\_RED, OUTPUT);

setup\_wifi();

client.setServer(mqtt\_server, 1883);

client.setCallback(callback);

dht.setup(DHT\_PIN, DHTesp::DHT22);

}

void loop() {

if (!client.connected()) reconnect();

client.loop();

unsigned long now = millis();

if (now - lastMsg > 2000) {

lastMsg = now;

auto data = dht.getTempAndHumidity();

client.publish("IOT/Test1/temp", String(data.temperature, 2).c\_str());

client.publish("IOT/Test1/hum", String(data.humidity, 1).c\_str());

}

}

**VII. VISUALISASI DAN DASHBOARD**

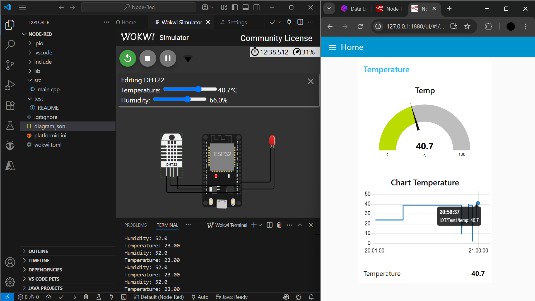
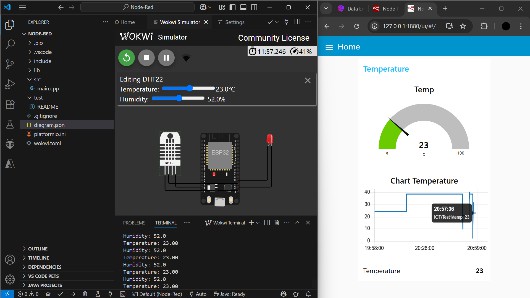
**Langkah Konfigurasi Node-RED:**

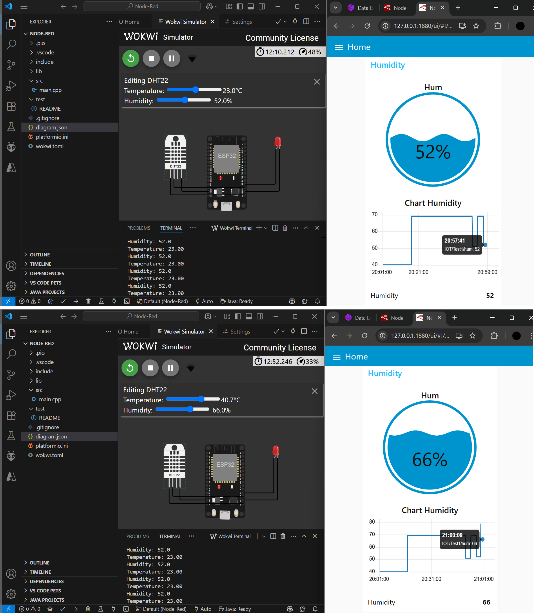
1. Install Node-RED dan node-red-dashboard.
2. Tambahkan MQTT IN node dan sambungkan dengan broker broker.emqx.io.
3. Buat node gauge dan chart untuk menampilkan suhu dan kelembaban.
4. Tambahkan InfluxDB Out untuk menyimpan data historis.
5. Buat dua flow untuk topik:
   * IOT/Test1/temp
   * IOT/Test1/hum
6. Deploy dan akses dashboard di http://127.0.0.1:1880/ui.

**VIII. HASIL SIMULASI**

**Data Output dari Serial Monitor:**

* **Suhu: 23.0 °C – 40.7 °C**
* **Kelembaban: 52.0% – 66.0%**
* **LED merespons perintah dari broker MQTT dengan benar**





# KESIMPULAN

Hasil praktikum monitoring suhu dan kelembaban menunjukkan bahwa dashboard dapat mengirimkan data secara realtime dari folder project baik dari wokwi maupun Visual Studio Code ke dashboard tampilan yang sudah diatur diagram alurnya dan konfigurasi setiap bagian. Dahboard terbagi menjadi dua halaman yakni halamn untuk IOT/Temperature dan halaman Humidity, keaduanya memiliki alur pengejerjaan yang sama tetapi yang membedakan adalah topic yang dipilih berbeda dan untuk elemen lainnya disesuaikan sesuai kebutuhan masing- masing